95-702 Distributed Systems For Information Systems Management

Project 3

Assigned: Friday, October 4, 2019
Due: Friday, Due Friday October 18, 11:59pm

Principles

One of our primary objectives in this course is to make clear the fundamental distinction between functional and nonfunctional characteristics of distributed systems. The functional characteristics describe the business or organizational purpose of the system. The non-functional characteristics affect the quality of the system. Is it fast? Does it easily interoperate with others? Is it fault tolerant? Is it reliable and secure?

In this project, we will illustrate an important nonfunctional characteristic of blockchain technology – its tamper evident design. We will build a stand-alone blockchain and a remote client that interacts with a blockchain API.

The student should note that this is not all of blockchain. There is more to learn. Real blockchains include peer to peer communication and many replicas of the blockchain. The blocks themselves typically include Merkle Trees. This assignment will certainly provide a nice foundation to build on.

Overview

In Task 0, you will write a blockchain by carefully following the directions in Javadoc format found here:

http://www.andrew.cmu.edu/course/95-702/examples/javadoc/index.html

The Javadoc describes two classes that you need to write – Block.java and BlockChain.java.

In Task 1, you will distribute the application that you created in Task 0. You will write a client server application. The interaction between the client and the server will be with JSON messages over TCP sockets. Thus, your work from Project2, Task 5 will be very useful and may be reused here.

For each task below, you must submit screenshots that demonstrate your programs running. These screenshots will aid the grader in evaluating your project.

Documenting code is also important. Be sure to provide comments in your code explaining what the code is doing and why. Be sure to separate concerns when

appropriate. You may include the Javadoc comments (provided) in your own code. But do comment on any additions or modifications that you make.

Task 0 Execution

Write a solution to Task 0 by studying the Javadoc provided on the course schedule. The logic found in Task 0 will be reused in Task 1.

The execution of Task 0, a non-distributed stand-alone program, will look almost exactly like the following interaction. You will select the exact same options and enter the exact same transactions. The only differences, of course, will be those due to the dynamic computations associated with the blockchain. As part of the submission of Task 0, you must turn in a screen shot such as the following:

```
run:
0. View basic blockchain status.
1. Add a transaction to the blockchain.
2. Verify the blockchain.
3. View the blockchain.
4. Corrupt the chain.
5. Hide the Corruption by repairing the chain.
6. Exit
0
Current size of chain: 1
Current hashes per second by this machine: 1846198
Difficulty of most recent block: 2
Nonce for most recent block: 1154
Chain hash:
00BC4767DC821A3F4B64B42017228734251FF1C0FF70EDDF9A66DC2C1AC7EFD8
0. View basic blockchain status.
1. Add a transaction to the blockchain.
2. Verify the blockchain.
3. View the blockchain.
4. Corrupt the chain.
5. Hide the Corruption by recomputing hashes.
6. Exit
2
Verifying entire chain
Chain verification: true
Total execution time required to verify the chain was 0 milliseconds
0. View basic blockchain status.
1. Add a transaction to the blockchain.
```

2. Verify the blockchain.
3. View the blockchain.
4. Corrupt the chain.
5. Hide the Corruption by recomputing hashes.
6. Exit
3
View the Blockchain
{"ds_chain" : [{"index" : 0,"time stamp " : "2019-02-22 17:22:14.133","Tx ":
"Genesis","PrevHash" : "","nonce" : 1154,"difficulty": 2}
],
"chainHash":"00BC4767DC821A3F4B64B42017228734251FF1C0FF70EDDF9A66DC
2C1AC7EFD8"}
0. View basic blockchain status.
1. Add a transaction to the blockchain.
2. Verify the blockchain.
3. View the blockchain.
4. Corrupt the chain.
5. Hide the Corruption by recomputing hashes.
6. Exit
1
Enter difficulty > 0
5
Enter transaction
Alice pays Bob 100 USD
Total execution time to add this block was 28071 milliseconds
0. View basic blockchain status.
1. Add a transaction to the blockchain.
2. Verify the blockchain.
3. View the blockchain.
4. Corrupt the chain.
5. Hide the Corruption by recomputing hashes.
6. Exit
1
Enter difficulty > 0
5
Enter transaction
Bob pays Eve 50 USD
Total execution time to add this block was 2066 milliseconds
0. View basic blockchain status.
1. Add a transaction to the blockchain.

2. Verify the blockchain.
3. View the blockchain.
4. Corrupt the chain.
5. Hide the Corruption by recomputing hashes.
6. Exit
1
Enter difficulty > 0
6
Enter transaction
Eve pays Charlie 10 USD
Total execution time to add this block was 47166 milliseconds
0. View basic blockchain status.
1. Add a transaction to the blockchain.
2. Verify the blockchain.
3. View the blockchain.
4. Corrupt the chain.
5. Hide the Corruption by recomputing hashes.
6. Exit
2
Verifying entire chain
Chain verification: true
Total execution time required to verify the chain was 0 milliseconds
0. View basic blockchain status.
1. Add a transaction to the blockchain.
2. Verify the blockchain.
3. View the blockchain.
4. Corrupt the chain.
5. Hide the Corruption by recomputing hashes.
6. Exit
3
View the Blockchain
{"ds_chain" : [{"index" : 0,"time stamp " : "2019-02-22 17:22:14.133","Tx ": "Genesis","PrevHash" : "","nonce" : 1154,"difficulty": 2},
{"index" : 1,"time stamp " : "2019-02-22 17:27:21.442","Tx ": "Alice pays Bob 100 USD","PrevHash" : "00BC4767DC821A3F4B64B42017228734251FF1C0FF70EDDF9A66DC2C1AC7EFD8","nonce" : 1457348,"difficulty": 5},
{"index" : 2,"time stamp " : "2019-02-22 17:28:27.419","Tx ": "Bob pays Eve 50 USD","PrevHash" :

"0000097BCBDF5146BAA13491D52B32946BE9ED8A7F0262EBC47A9AA7213E85F7","nonce" : 107856,"difficulty": 5},
{"index" : 3,"time stamp " : "2019-02-22 17:29:16.839","Tx ": "Eve pays Charlie 10 USD","PrevHash" :
"0000017BA1238C419C992654EAFC5417F1BDBAF2FECFCF44328B0CCD777EDBF4","nonce" : 2475995,"difficulty": 6}
],
"chainHash":"000000B2594B8EE49149604B87835990E6EF36C2371768A13A36B2F1EE2EF3EF"}
0. View basic blockchain status.
1. Add a transaction to the blockchain.
2. Verify the blockchain.
3. View the blockchain.
4. Corrupt the chain.
5. Hide the Corruption by recomputing hashes.
6. Exit
0
Current size of chain: 4
Current hashes per second by this machine: 1946252
Difficulty of most recent block: 6
Nonce for most recent block: 2475995
Chain hash:
000000B2594B8EE49149604B87835990E6EF36C2371768A13A36B2F1EE2EF3EF
0. View basic blockchain status.
1. Add a transaction to the blockchain.
2. Verify the blockchain.
3. View the blockchain.
4. Corrupt the chain.
5. Hide the Corruption by recomputing hashes.
6. Exit
4
Corrupt the Blockchain
Enter block ID of block to Corrupt
2
Enter new data for block 2
Charlie pays Dave 3 USD
Block 2 now holds Charlie pays Dave 3 USD
0. View basic blockchain status.
1. Add a transaction to the blockchain.
2. Verify the blockchain.

3. View the blockchain.
4. Corrupt the chain.
5. Hide the Corruption by recomputing hashes.
6. Exit
3
View the Blockchain
{"ds_chain" : [{"index" : 0,"time stamp " : "2019-02-22 17:22:14.133","Tx ": "Genesis","PrevHash" : "","nonce" : 1154,"difficulty": 2},
{"index" : 1,"time stamp " : "2019-02-22 17:27:21.442","Tx ": "Alice pays Bob 100 USD","PrevHash" : "00BC4767DC821A3F4B64B42017228734251FF1C0FF70EDDF9A66DC2C1AC7EFD8","nonce" : 1457348,"difficulty": 5},
{"index" : 2,"time stamp " : "2019-02-22 17:28:27.419","Tx ": "Charlie pays Dave 3 USD","PrevHash" : "0000097BCBDF5146BAA13491D52B32946BE9ED8A7F0262EBC47A9AA7213E85F7","nonce" : 107856,"difficulty": 5},
{"index" : 3,"time stamp " : "2019-02-22 17:29:16.839","Tx ": "Eve pays Charlie 10 USD","PrevHash" : "0000017BA1238C419C992654EAFC5417F1BDBAF2FECFCF44328B0CCD777EDBF4","nonce" : 2475995,"difficulty": 6}
],
"chainHash":"000000B2594B8EE49149604B87835990E6EF36C2371768A13A36B2F1EE2EF3EF"}
0. View basic blockchain status.
1. Add a transaction to the blockchain.
2. Verify the blockchain.
3. View the blockchain.
4. Corrupt the chain.
5. Hide the Corruption by recomputing hashes.
6. Exit
2
Verifying entire chain
..Improper hash on node 2 Does not begin with 00000
Chain verification: false
Total execution time required to verify the chain was 0 milliseconds
0. View basic blockchain status.
1. Add a transaction to the blockchain.
2. Verify the blockchain.
3. View the blockchain.
4. Corrupt the chain.

5. Hide the Corruption by recomputing hashes.
6. Exit
5
Repairing the entire chain
Total execution time required to repair the chain was 26132 milliseconds
0. View basic blockchain status.
1. Add a transaction to the blockchain.
2. Verify the blockchain.
3. View the blockchain.
4. Corrupt the chain.
5. Hide the Corruption by recomputing hashes.
6. Exit
3
View the Blockchain
{"ds_chain" : [{"index" : 0,"time stamp " : "2019-02-22 17:22:14.133","Tx ": "Genesis","PrevHash" : "","nonce" : 1154,"difficulty": 2},
{"index" : 1,"time stamp " : "2019-02-22 17:27:21.442","Tx ": "Alice pays Bob 100 USD","PrevHash" : "00BC4767DC821A3F4B64B42017228734251FF1C0FF70EDDF9A66DC2C1AC7EFD8","nonce" : 1457348,"difficulty": 5},
{"index" : 2,"time stamp " : "2019-02-22 17:28:27.419","Tx ": "Charlie pays Dave 3 USD","PrevHash" : "0000097BCBDF5146BAA13491D52B32946BE9ED8A7F0262EBC47A9AA7213E85F7","nonce" : 441725,"difficulty": 5},
{"index" : 3,"time stamp " : "2019-02-22 17:29:16.839","Tx ": "Eve pays Charlie 10 USD","PrevHash" : "00000B1C711B1D81B7724D0FC11C3D2AC51345E505B4CF72121D246519C20101","nonce" : 852425,"difficulty": 6}
],
"chainHash":"000000D44CB102756678DBE10C321D2A93F9FCA9A8FE370FF888C18CAF7C11D0"}
0. View basic blockchain status.
1. Add a transaction to the blockchain.
2. Verify the blockchain.
3. View the blockchain.
4. Corrupt the chain.
5. Hide the Corruption by recomputing hashes.
6. Exit
2
Verifying entire chain

Chain verification: true
Total execution time required to verify the chain was 0 milliseconds
0. View basic blockchain status.
1. Add a transaction to the blockchain.
2. Verify the blockchain.
3. View the blockchain.
4. Corrupt the chain.
5. Hide the Corruption by recomputing hashes.
6. Exit
6

Task 0 Grading Rubric 50 Points

See the Javadoc for a description of the main routine and the difficulty levels.

	Excellent	Good – minor problems	Poor – serious problems	No submission
Works. A screen scrape is provided that shows the exact same user selections.	44	40	30	0
The main method of the BlockChain class is documented and describes behavior with difficulty levels of 4 and 5 (see Javadoc)	4	3	2	0
Separation of concerns & good style	1	0	0	0
Submission requirements met	1	0	0	0

Task 1 Execution

The execution of Task 1 will appear exactly the same as in Task 0. The primary difference will be that, behind the scenes, there will be a client server interaction using JSON over TCP sockets.

Use Project 2, Task 5 as a guide. Each request to the server must be signed using the private key as was done in Project 2, Task 5. The signature must be checked on the server. If the signature fails to verify, send an appropriate error message back to the client.

You are required to design two JSON messages types – a message to encapsulate requests from the client and a message to encapsulate responses from the server. Each JSON request message will include a signature.

Task 1 Grading Rubric 50 Points

	Excellent	Good – minor problems	Poor – serious problems	No submission
This is a working client and server. A screen scrape is provided that shows the exact same user selections as in Task 0.	40	36	30	0
The format of the JSON request and response messages is designed well.	4	3	2	0
The server verifies the	4	3	2	1

signature on each request.				
Separation of concerns & good style	1	0	0	0
Submission requirements met	1	0	0	0

Project 3 Submission Requirements

Documentation is always required.

Remember to separate concerns.

The IntelliJ projects will be named as follows:

- Project3Task0 (You need to zip this folder)
- Project3Task1Server (You need to zip this folder)
- Project3Task1Client (You need to zip this folder)

You should also have two screen shot folders:

- Project3Task0-screenshot (Do not zip)
- Project3Task1-screenshot (Do not zip)

The submission should be a single zip file. This file will be called YourAndrewID.zip.

Submission file structure:

- YourAndrewID.zip
  - Project3Task0.zip
  - Project3Task0-screenshot
  - Project3Task1Server.zip
  - Project3Task1Client.zip
  - Project3Task1-screenshot